How to hybrid MPI/openMP on LCRC servers

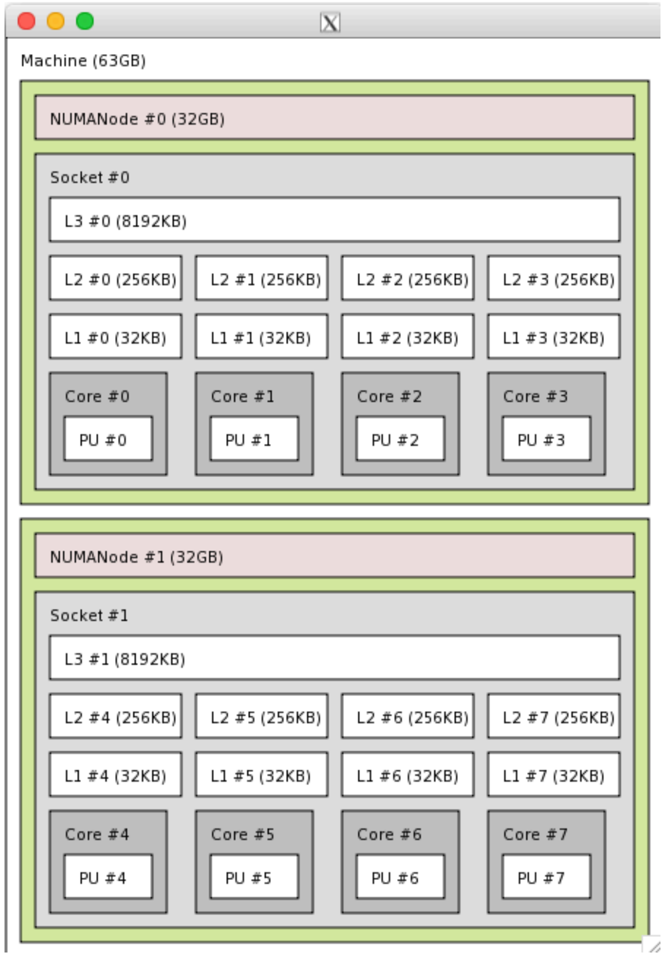
John J. Low

8/13/15

Overview

- Hybrid MPI/OpenMP applications can be used to
  - Decrease the needed memory
  - Improve load balance of MPI applications to improve performance.
- However most MPI/OpenMP applications are less efficient than pure MPI applications.
  - Multithreaded hybrid MPI/OpenMP are often need to get some parallel scaling for a thousand or more cores
- Running hybrid MPI/OpenMP applications is not trivial!
  - CPU affinity should be defined to maximize performance.
    - In general, all threads from a MPI process should share the same memory cache.
    - The best performing combination of MPI process and openMP threads will depend on the application.
      - Only experimentation will define the best combination.
- The commands needed to run a MPI/OpenMP applications will vary with MPI version and server.
  - The examples shown in this presentation are valid for mvapich2 versions 1.9 or higher.

Fusion



- This figure is output from the lstopo command on Fusion.
  - It graphically describes the topology of a fusion node.
  - Fusion how two processors with four cores each.
  - Cores 0-3 are located on Socket 1
  - Cores 4-7 are located on Socket 2
  - Cores on the same socket share the same L3 cache.
- Two or Four threads per mpi process will likely be most efficient.
  - They will share the same L3 cache if the CPU_Affinity is defined correctly.

Fusion

- Commands to run mpi/openmp programs on Fusion.
  - Set environmental variables with the following commands.

    ```
    export  MV2_ENABLE_AFFINITY=0
    export  OMP_NUM_THREADS=4
    ```

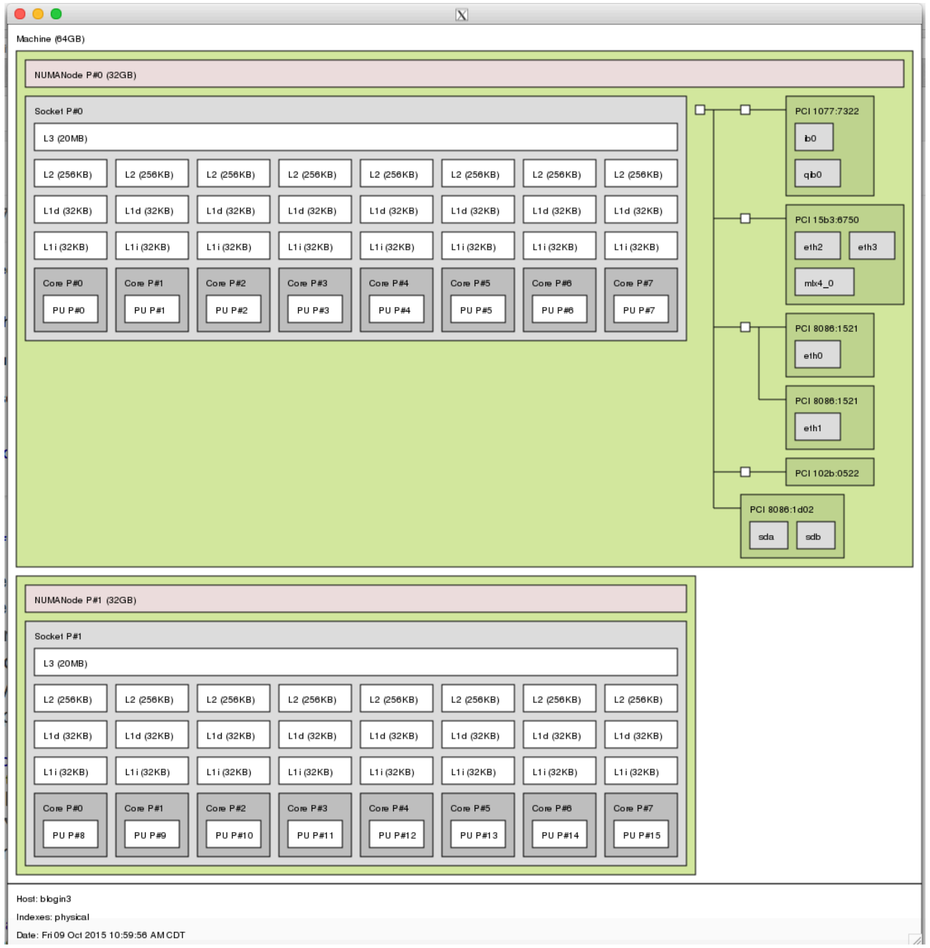
    - The above commands will turn off the default CPU affinity and set the number OpenMP threads to 4.
  - Set the “ppn” PBS attribute to the number of cores per node divided by the number of openMP threads in your PBS script with the following line in your PBS script.

    ```
    #PBS -l nodes=2:ppn=2
    ```

Fusion

- The following command will start two MPI processes with four OpenMP threads each.
  - mpirun -bind-to user:0+1+2+3,4+5+6+7 -np 2 a.out
- If you have assigned the environmental variables MV2_ENABLE_AFFINITY and OMP_NUM_THREADS as in the previous page then:
  - MPI process 1 and the threads it spawns will run on cores 0-3.
  - MPI process 2 and the threads it spawns will run on cores 4-7.
  - All the threads from a the same MPI process will use cores sharing the same L3 cache.
- The following command will start four MPI processes with two OpenMP threads each.
  - mpirun -bind-to user:0+1,2+3,4+5,6+7 –np 4 a.out
- Two or four OpenMP threads per process should be more efficient than eight threads per process, assuming a total of eight threads per node.
- You will have to experiment to determine whether two or four openmp threads per MPI process is more efficient for your application.

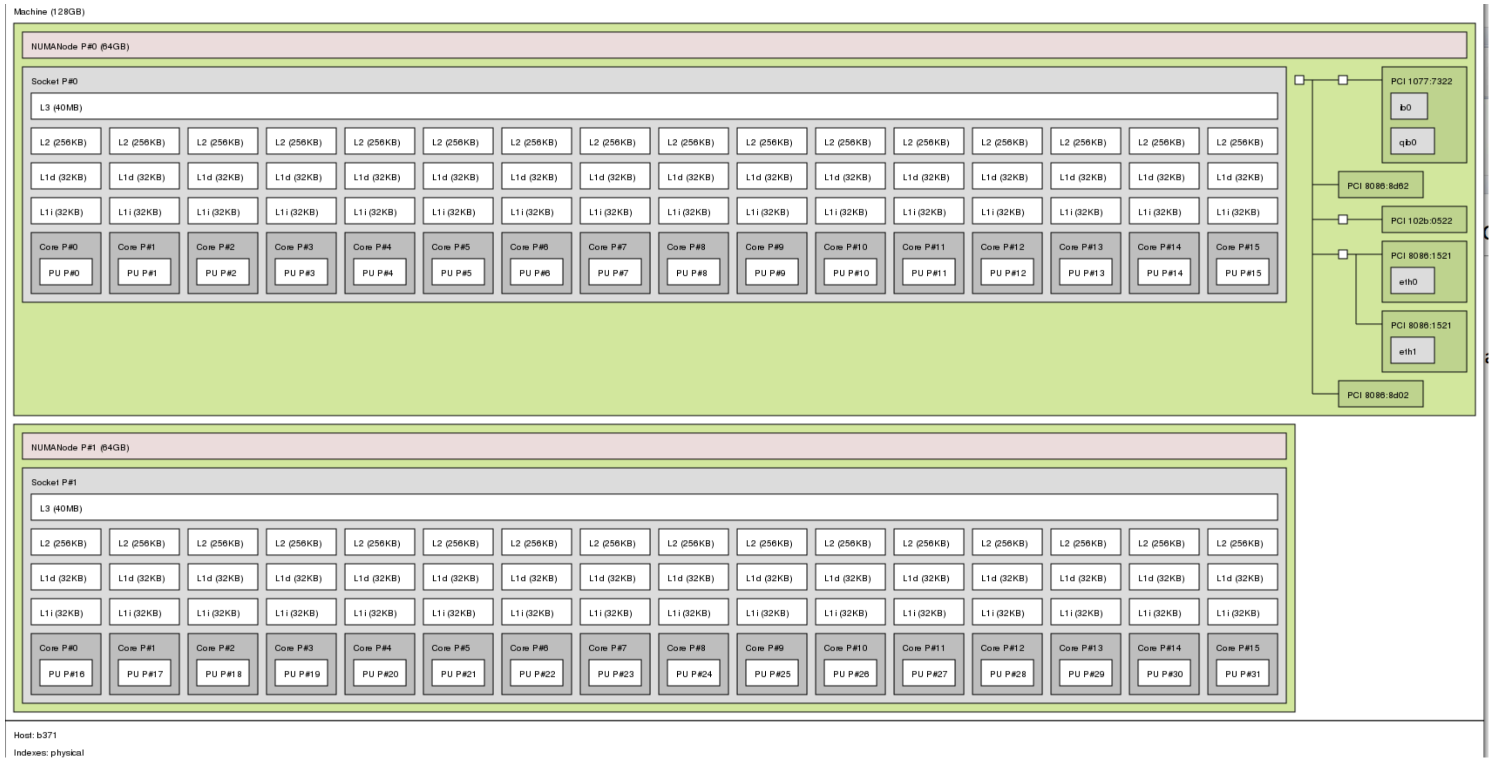
Blues – Sandy Bridge Nodes



- The output from lstopo shows that the "Sandy Bridge" nodes on blues have two processers with 8 cores each.
- All cores in the same processor share the same L3 cache.
- Eight, four or two openmp threads per MPI process should be efficient.
  - You have to experiment to determine which is most efficient for your application.
- To start two MPI processes with 8 openmp threads each, on each node:
  - Set the ppn PBS variable to 2
  - Set the OMP_NUM_THREADS to 8
    - export OMP_NUM_THREADS=8
- Use the following command to run your application.

```
mpirun -bind-to user:  \
0+1+2+3+4+5+6+7,8+9+10+11+12+13+14+15 \
-np 2 a.out
```

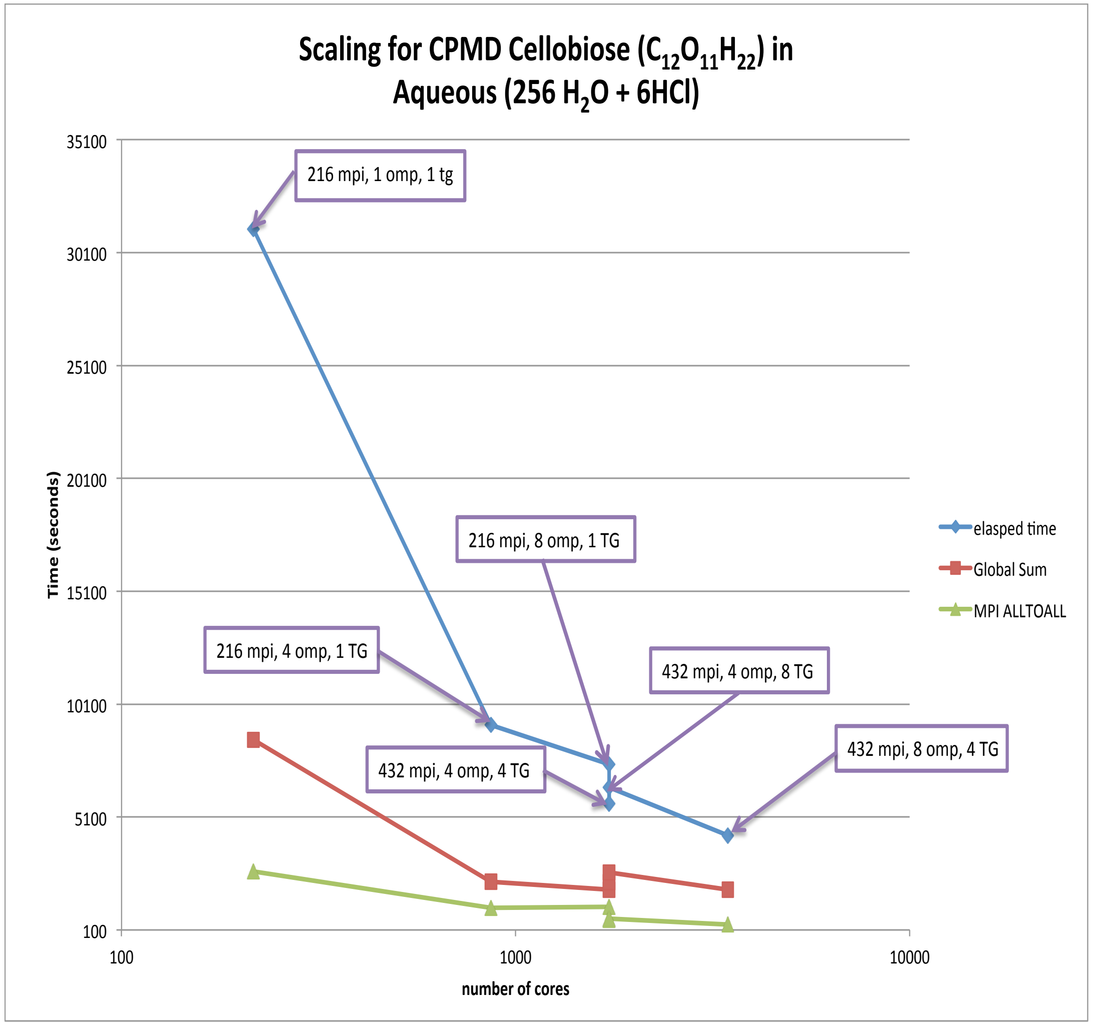
Blues – Haswell Nodes



- The output from lstopo shows that the Haswell nodes on blues have two processers with 16 cores each.
- All cores in the same processor share the same L3 cache.
- Eight, four or two openmp threads per MPI process should be efficient.
  - You have to experiment to determine which is most efficient for your application.
- To start two MPI processes with 16 openmp threads each, on each node:
  - Set the ppn PBS variable to 2
  - Set the OMP_NUM_THREADS to 16
    - export OMP_NUM_THREADS=16
- Use the following command to run your application.

```
mpirun -bind-to user:  0+1+2+3+4+5+6+7,8+9+10+11+12+13+14+15 -np 2 a.out
```

An Example - CPMD



CPMD is an example of a MPI/ OpenMP program.

- This benchmark was run on blues.
- 216 hybrid OpenMP/MPI processes with 4 threads each on 864 cores or 108 fusion nodes ran 3.4 times faster than 216 pure MPI processes (1 OpenMP thread per process) on 32 nodes of fusion.
- Note that 216 hybrid MPI/ OpenMP processes with 8 threads on 1728 cores ran only 23% faster than 216 hybrid MPI/OpenMP with 4 threads on 864 cores.